JARS Journal of Applied Remote Sensing

RESEARCH PAPER

Convolutional neural network-based crowd detection for COVID-19 social distancing protocol from unmanned aerial vehicles onboard camera

Leonard Matheus Wastupranata* **and Rinaldi Munir**
Institut Teknologi Bandung, School of Electrical Engineering and Informatics, Bandung, Indonesia

ABSTRACT. Social distancing is a feasible solution to break the chain of the spread of coronavirus disease 2019 (COVID-19). A human crowd detection model was trained with a computational load that can be handled by a companion computer on the unmanned aerial vehicle (UAV) to minimize the spread of COVID-19. The model is designed to be able to measure social distance between people, whether it exceeds predetermined safe limits (1.5 m). The convolutional neural network model was trained using a dataset of 9600 images featuring humans, cyclists, and motorcyclists, with an allocation of 200 images each for testing and hyperparameter tuning. The image dataset was extracted from videos recorded above the UAV in the Institut Teknologi Bandung area, capturing diverse crowd scenarios throughout the day. The pretrained model for transfer learning method is a single shot detector with MobileNet, ResNet50, and ResNet101 architectures. The measurement of the estimated social distance uses the Euclidian distance with the average Indonesian human as a reference, which is 1.6 m. MobileNet V2 was chosen as a crowd detection model with a lightweight size, which is only 19 MB and the average detection runtime for a single image is only 0.606s, in accordance with the load for the onboard companion computer. MobileNet V2 is also able to detect crowds of people well with the precision value reaching 84.9% and the recall value reaching 87.8%.

 [DOI: 10.1117/1.JRS.17.044502]

Keywords: calibration; COVID-19; human detection; social distance estimation; unmanned aerial vehicles

Paper 230076G received Mar. 1, 2023; revised Aug. 4, 2023; accepted Sep. 26, 2023; published Oct. 9, 2023.

1 Introduction

The coronavirus disease 2019 (COVID-19) pandemic has swept across the world and changed the way people live in general. The spread of this virus is extremely fast and massive, especially in environments that are crowded with people. The World Health Organization has suggested that steps to anticipate the spread of the virus, including close contact and crowds, should be minimized by implementing social distancing.[1] Several studies have proven that implementing social distancing has lowered the risk of spreading the virus and reduced mortality rates that may arise.[2–5] Crowd is a condition that occurs when two or more people caught on camera are close together at <1.5 m.[6] Crowd detection is a research topic regarding the observation of a large collection of people who violating social distance in a certain area.

In crowd detection, a sensor is needed that can capture data to be translated into other forms into information on the state of the crowd. The information can be received in various forms, such as the result of human enumeration in a crowd,[7] geographical location in a density map,[8] and

*Address all correspondence to Leonard Matheus Wastupranata, leo.matt.547@gmail.com

1931-3195/2023/$28.00

estimation of social distance between humans and each other.[9] Computer vision is a specific branch of science that extracts a digital image, then produces information that can be processed into several methods, such as counting methods, measuring distances, or navigation.[10] On the other hand, convolutional neural network (CNN) is a type of neural network that forward signals into a stack of convolutional layers. The output of the last layer will be spread into a stack called the fully-connected layer.[11] Currently, crowd detection using CNN and computer vision is the best solution to anticipate close contact between people in a crowd.

Crowd detection can be achieved by installing multiple static cameras at strategic crowd locations.[12] To overcome occlusions, cameras placed at a high elevation track only people's heads, and by combining data from multiple views, height information is extracted and used for head segmentation, detecting head tops as two-dimensional (2D) patches at various heights through intensity correlation applied to aligned frames from different cameras. The deployment of multiple cameras with overlapping fields of view facilitates robust tracking of individuals in densely populated scenes. However, these cameras have limitations in ensuring comprehensive coverage of all monitored areas where crowds may gather. In addition, the maintenance and repair costs of multiple static cameras, which are susceptible to potential damage, can be substantial.[12,13] Therefore, the utilization of unmanned aerial vehicles (UAVs) becomes necessary to extend the detection range of human crowds and mitigate occlusion issues that arise when multiple human objects are stacked upon one another.

Research on crowd detection in relation to social distancing monitoring has been conducted with different results and conditions. Papaioannidis et al.[14] have created an image segmentation model that can detect crowds to determine the safe flying altitude of UAV's with accuracy rate ranging from 85% to 98%. However, the image segmentation results cannot be used to estimate social distance. Rezaee et al.[15] have also trained an image segmentation model for each case of contact between humans using UAVs. The accuracy of detection reaches 97.5% of the 100% scale, but the social distancing violation model is displayed in the same detection box and the detection is not able to estimate the distance between people. Another approach to crowd detection was designed by Shao et al.,[16] which detects pedestrians using UAVs and transforming human head images. The detection accuracy reaches 88.5% for video processing with 75 FPS. However, there is no indication of the specific location where a violation occurred, either in the form of contact lines between people or the entire human image.

The objective of this paper is to create a CNN model that can detect crowds of people from the point of view of the UAV camera. The main goal is to build a lightweight model so the computational process is below the memory capacity of the companion computer in the UAV with high precision and recall values. Next, the new CNN model will measure the estimated distance between humans who are close to each other, with the help of computer vision. From the results of the estimated social distance measurement, a program will be developed to display the counting of social distancing violations between two or more humans who are close together.

This paper is divided into four parts, starting with an introductory section that explains the background of this research and related works that explain the crowd detection research that was previously researched. Then, the results and discussion section will discuss the human crowd detection model testing output and its analysis. Finally, there are conclusions and future works for the further development of this research.

2 Related Works

The approach of crowd detection with hidden Markov models (HMMs) simulations, as presented by Butenuth et al.,[17] focuses on detecting the density of crowds. However, it should be noted that using simulation models such as HMMs may not necessarily lead to generalizable results. HMMs often make assumptions about linear data distributions, which may not capture the complexity of real-world crowd scenarios.

In contrast, deep learning algorithms, such as CNNs, have gained popularity due to their ability to model complex data and learn hierarchical representations.[18] Deep learning models can capture intricate patterns and relationships in crowd scenes, enabling them to handle non-linear data distributions more effectively compared to traditional HMMs.

Human detection can be performed using histograms of oriented gradient (HOG) features.[19–21] However, numerous poses can be formed due to the flexibility of the human body, which can hinder the process of human detection. Choi et al.[22] demonstrated that human detection accuracy using CNN achieved an accuracy of more than 80%, which is higher than the accuracy achieved using HOG methods. Therefore, CNNs are required to better understand the human objects involved in a crowd.[23]

There are two object detection approaches, which are one-stage object detection and two-stage object detection. The one-stage object detection approach was chosen because high detection inference speed was prioritized.[24] Models that belong to the one-stage object detection approach are single shot detection (SSD), RetinaNet, EfficientDet, and others. SSD architecture is based on a feed-forward convolution network approach. A collection of bounding boxes with a fixed size along with their values will be generated to predict the existence of the object class in the box, followed by a non-maximum suppression step to generate predictions at the final detection stage.[25] The main difference between SSD architecture training methods and other detection architectures is that there is no need for a region proposal on the SSD, only a small convolution filter is needed. This filter will be run after the feature mapping layer in the convolution stage to get the class prediction results. Therefore, detection processing can be conducted quickly, and the resulting detection box will be more accurate.

Human crowd detection models have been generated in the study of Papaioannidis et al.[14] as a sensor for determining the safe flight altitude of a UAV. The method used to perform the detection is a training model based on image segmentation on a convoluted neural network. The experiment was conducted by inserting the model into the embedded system. This model has a high detection accuracy, which is at a confidence value of 85% to 98% depending on the size of the image to be processed. The disadvantage of this study is that it can only detect crowds at general locations and is only intended as a UAV flight altitude determination device. Unfortunately, this model is not suitable for detecting humans at close range and cannot capture detailed images of human objects.

Another crowd model training was conducted by Rezaee et al.[15] using ShuffleNet, an image segmentation model to capture human objects and create a detection box for humans who are caught in close proximity to each other. Humans will be detected using the Kalman filter method to track human movements from above the UAV. The accuracy obtained is quite high, which is around 97.5% with an average processing time of about 84 ms for each video frame. However, the model cannot calculate the distance between two or more captured human objects. This is because in image segmentation, the coordinate center of one human object cannot be determined from another human object. In addition, the resulting frames per second (FPS) is also quite low, it can only process videos with a size of 11 FPS. The optimization is needed to improve the performance of the related human detection model.

Another approach to detecting crowds of people from above the UAV is to detect human head images and perform transformations so that they are like the representation of location coordinates on a 2D matrix.[16] This model is trained using PeleeNet with a high level of precision, which is about 88.22% in video processing at 76 FPS. In experiments using UAV, the level of precision obtained is about 88.5% in video processing of 75 FPS. This model can also be applied to the human object counting system for those who violate social distancing rules. Unfortunately, the model cannot capture the entire human body image. As a result, the results of image extraction will only get human heads and make identification of violators difficult. In addition, there are no markers where the social distancing violations occurred, such as lines or other signs. Therefore, a complete human detection is needed and can display the location of the violation in the image set to be processed.

The field of crowd detection and human counting encompasses several complex challenges that demand scientific investigation and resolution. First, the diverse range of human body sizes presents a significant obstacle in achieving reliable detection and precise counting across individuals of varying proportions. In addition, the presence of visual noise and the ability to accurately distinguish humans from similar objects within crowded environments necessitates the application of advanced feature extraction and classification techniques. Furthermore, ensuring computational efficiency is of utmost importance for real-time implementation, given the large number of individuals typically present in crowd scenes that require rapid processing.

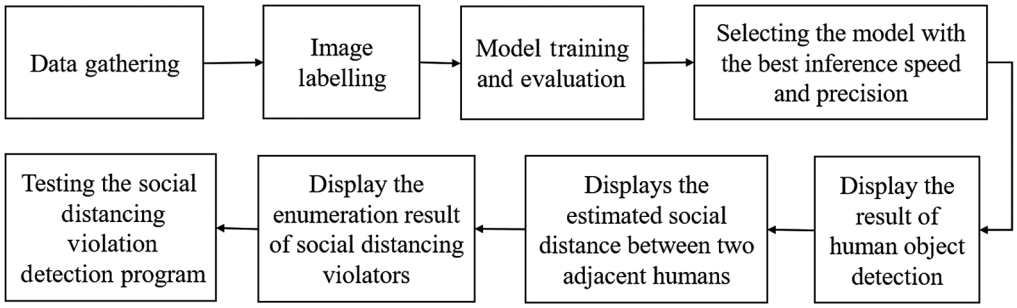


Fig. 1 Proposed workflow for crowd detection system.

Moreover, the generalization of detection and counting algorithms to diverse environments is hindered by factors such as variations in lighting conditions, background changes, and camera perspectives,[26] thereby necessitating the development of robust and adaptable models to overcome these challenges.

3 Proposed Method

3.1 Proposed Workflow

In the data gathering process (as shown in Fig. 1), the video containing human detection will be extracted into a single image set, each of which will be labeled. Next, a model training and evaluation process will be conducted to produce a model that can adapt to the human crowd image captured from the UAV camera. After that, the model that could detect human objects precisely with the fastest inference time will be selected. Then, the program will be developed to process the model's detection results into a detection box that can be displayed to the screen using computer vision. The process of counting camera-captured social distancing violators and storing captured images of social distancing violations will be managed by an algorithm developed in Python. Both the enumeration results and the captured images will be stored in a log in a specific folder that can also be accessed by the user. Finally, an overall program test will be conducted to ensure that the detection program runs well.

3.2 System Architecture Design

When developing a whole crowd detection program, there is a system that helps few processes (as shown in Fig. 2). In the initial stage, the video obtained in real-time from the camera on the UAV will be adjusted to FPS at certain frame intervals. After obtaining the image capture in a frame, the human object detection process will then be conducted. The detection model that will be used has previously been stored in the internal storage embedded in the UAV's companion computer to make processing the detection results easier. The next step is to get all the detection boxes contained in the captured image of the detection results, the detection box to be processed

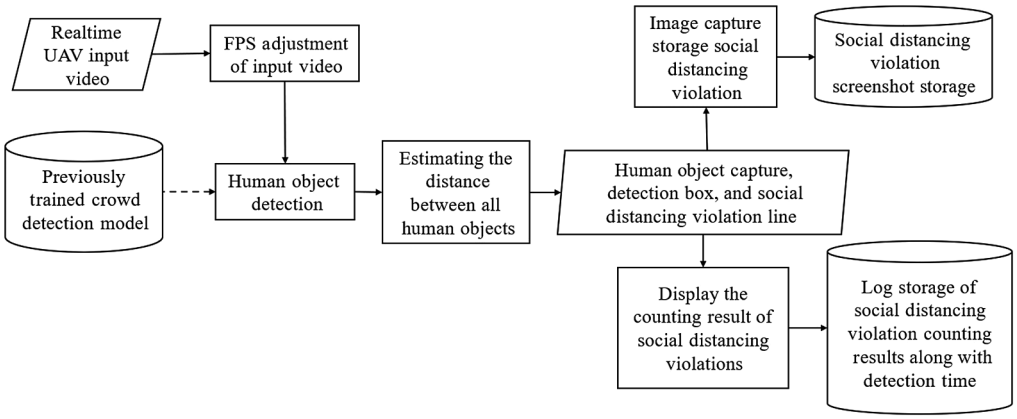


Fig. 2 System architecture design for crowd detection.

is the classification of objects that indicate the "human" class. By estimating the distance between all detection boxes, a human object that has violated social distance will be obtained in the form of specific coordinates of the detection box and the estimated number of social distance violations for each detection box. The data will be processed to produce a capture of human objects, detection boxes, and social distance violation lines.

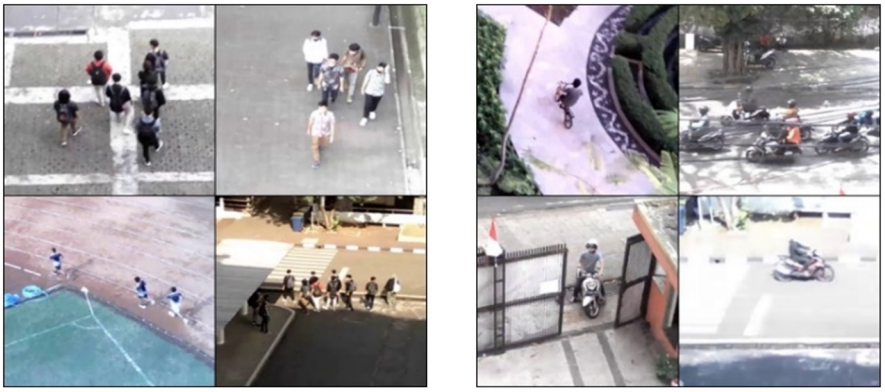
For the social distancing violation capture image, an image containing two human object violators and one social distancing violation line will be produced. The more detailed image processing results will be stored in the internal storage. On the other hand, the social distancing violation line data will be collected and then the number of lines generated will be enumerated. The number of lines represents the social distancing violations that occur in one image capture at a given time. The results of this enumeration will be written into a file containing a log of the number of social distancing violation events along with the specific time of the event. This log will be stored in the internal storage dedicated to the number of social distancing violations. The entire process previously described will run continuously until the UAV stops operating. The stop condition occurred when the state of the UAV after getting the landing command and returning to Home.

3.3 Datasets

There are several image datasets that can be accessed and used publicly. However, the convolution model tends to have difficulty recognizing different detection environments from the training data. Some pre-trained models can detect humans well, but it is difficult to detect humans from the point of view of the UAV camera. Therefore, it is necessary to independently adjust the image dataset by adjusting the training data taken from the camera when the UAV is flying.

Images that represent human objects will be collected to conduct model training. Image data retrieval process uses a video extraction process for each frame to be saved in portable network graphic (PNG) format. PNG format was chosen because the compression type in this format is lossless. This format can improve the accuracy of object detection at the representation of small pixels.[27] Video recording of the crowd is generated by the UAV camera in flight, at an altitude of about 5 to 10 m from the ground. Positive dataset [can be seen in Fig. 3(a)] contains human objects when standing or walking and is categorized as a "human" class. On the other hand, negative dataset [can be seen in Fig. 3(b)] contains objects such as cyclists or motorcyclists which are categorized as "non_human" class. Therefore, the amount of positive data with negative data is balanced so that there is no oversampling in certain classes.

Videos containing crowds [Institut Teknologi Bandung (ITB) area] were recorded in the morning to evening timeframe so the camera could capture images with bright and clear conditions. After validating the objects contained in the image, 10,000 images from video extraction in sizes of 640×480 pixels have been collected. Datasets that have been collected will be labeled according to their respective class categories using the labelImg tool.[28] The output of the image data labeling process is in Extensible Markup Language (XML) format following the PASCAL visual object classes convention.[29] Furthermore, the image datasets that have been assigned class category labels will be divided into three folders, more details can be seen in Table 1.



(a) (b)

Fig. 3 Example of (a) positive dataset and (b) negative dataset.

Table 1 Dataset splitting.

Folder	Positive images	Negative images	Total images
Train	4800 (48%)	4800 (48%)	9600 (96%)
Dev	100 (1%)	100 (1%)	200 (2%)
Test	100 (1%)	100 (1%)	200 (2%)
Total	5000 (50%)	5000 (50%)	10,000 (100%)

Table 2 SSD pre-trained model in TensorFlow model zoo.[30]

Model name	Speed (ms)	Common objects in context dataset mean average precision (COCO mAP)	Output
SSD MobileNet v2 320 × 320	19	20.2	Boxes
SSD MobileNet V1 FPN 640 × 640	48	29.1	Boxes
SSD MobileNet V2 FPNLite 320 × 320	**22**	**22.2**	**Boxes**
SSD MobileNet V2 FPNLite 640 × 640	39	28.2	Boxes
SSD ResNet50 V1 FPN 640 × 640 (RetinaNet50)	**46**	**34.3**	**Boxes**
SSD ResNet50 V1 FPN 1024 × 1024 (RetinaNet50)	87	38.3	Boxes
SSD ResNet101 V1 FPN 640 × 640 (RetinaNet101)	**57**	**35.6**	**Boxes**
SSD ResNet101 V1 FPN 1024 × 1024 (RetinaNet101)	104	39.5	Boxes
SSD ResNet152 V1 FPN 640 × 640 (RetinaNet152)	80	35.4	Boxes

3.4 Transfer Learning from Pre-Trained Model

For the selection phase of the pre-trained model, considerations are made from the side of the most optimal speed with high enough precision (Table 2, which have bold font). SSD MobileNet V2 lightweight feature pyramid network (FPNLite) is selected because the model provides a high detection speed and a better level of precision than other MobileNet models. With the addition of this FPNLite feature, objects with small sizes will be able to be detected better than the standard MobileNet V2 model.[31] Moreover, the SSD ResNet50 and the SSD ResNet101 were chosen with a layer size of 640 × 640 compared to 1024 × 1024 because they fit the needs of model training and testing. The SSD ResNet152 was not selected due to the excessive number of layers and the final model memory being inefficient for the case of crowd detection using the UAV companion computer.

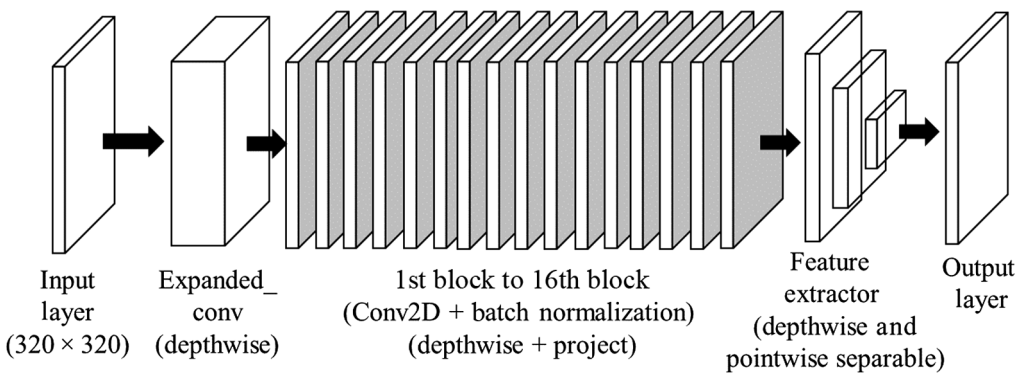


Fig. 4 Layer of SSD MobileNet V2 FPNLite 320 × 320.

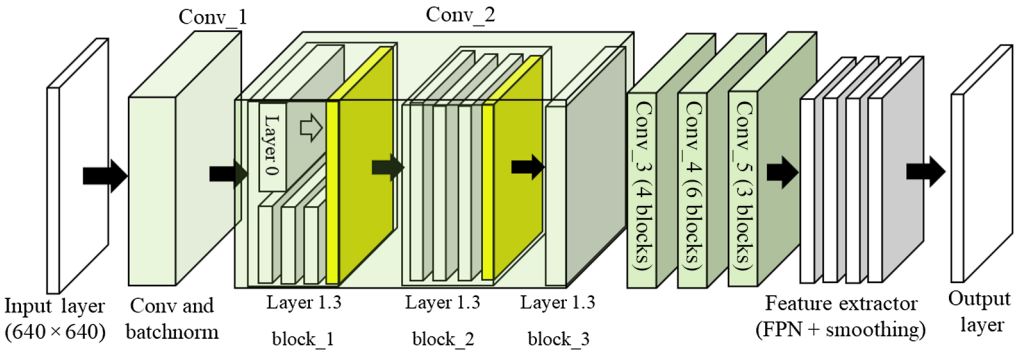


Fig. 5 Layer of SSD ResNet50 V1 feature pyramid network (FPN) 640 × 640 (RetinaNet50).

In the MobileNet architecture (as shown in Fig. 4), the very first layer will receive inputs with dimensions of 320 × 320. After that, there is an additional convolution which is the base layer of the SSD architecture. Furthermore, the output of the additional convolution layer will enter 16 blocks, which are useful for depthwise separable process along with batch normalization process. This normalization is useful for scaling the output of each previous layer so the input to the convolutional layer afterward becomes more adaptive. Finally, depthwise and pointwise feature processing will be performed. For depthwise, the process is performed at a kernel size of 3 × 3 while at pointwise, the kernel size used is simply 1 × 1 dimension.

In the ResNet50 architecture, the input layer dimension size is 640 × 640, and will be processed in conv_1 as the base layer of the SSD architecture. Furthermore, the output of conv_1 will enter conv_2. In the conv_2 section, there are three main blocks, and each block has three layers. In the first block, there is a Layer_0 that will directly send the results to the temporary output in the same block (can be seen in Fig. 5 in the yellow layer). This is in accordance with the residual learning principle on operations within a particular block. For other blocks, the process runs sequentially like any other process. After all blocks have been processed, the output of the convolution will be input to the next convolution process. While the ResNet convolution process has been completed, the final feature extraction and normalization process will be conducted to determine the class classification along with the location of the detection box. The result of this process will be forwarded to the output layer.

The layer architecture in ResNet101 is similar to ResNet50 (can be seen in Fig. 6). First, the input data are received with dimensions of 640 × 640 and will be processed in conv_1 as the base layer of SSD architecture. Next, the residual learning system will be applied to block_1 in Layer_0 whose output will be stored while waiting for other block processing. The number of blocks in the convolution layer is more than the ResNet50 model. As seen in Fig. 6, the number of blocks in conv_4 is 23 units, more than the ResNet50 model (only 6 blocks). After the convolution process is complete, the feature extraction and final normalization process is conducted to determine the class classification and the location of the detection box.

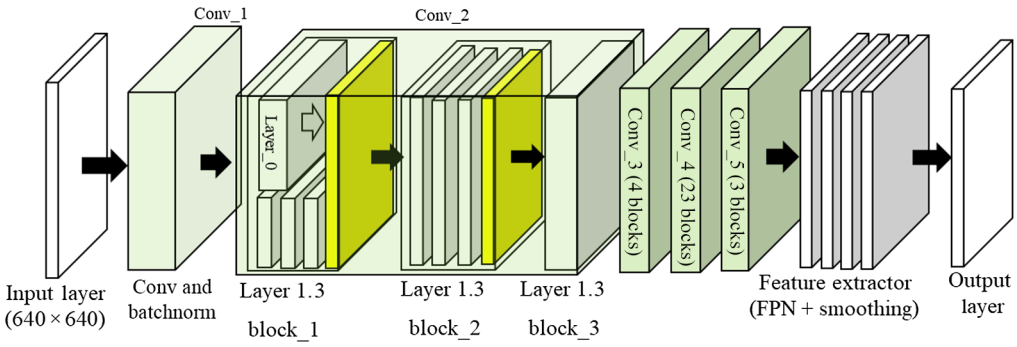


Fig. 6 Layer of SSD ResNet101 V1 FPN 640 × 640 (RetinaNet101).

3.5 Hyperparameter Tuning

For all pre-trained models, hyperparameter adjustments will be made to fit the training process on a larger dataset. For tuning phase, development dataset (about 200 images) will be used to support the best-fit method armed with experience from numbers obtained in related research.[32,33] There are several hyperparameters that need to be tuned, such as learning rate base, cosine decay step,[34–36] exponential decay step,[37] warmup learning rate,[38,39] and warmup step.[40] The best combination of hyperparameters is expected to increase the speed of the training process on a larger amount of data without worrying about the precision and sensitivity of the model.[41,42] Hyperparameter tuning is using graphics processing unit (GPU) computing environment with a total epoch of 100,000 steps. The hyperparameter combination in each model is selected based on the lowest total loss value, the highest precision value, and the highest sensitivity (recall) value, as can be seen in Table 3 with bold font.

3.6 Ratio of Social Distance to Human Height

A digital image consists of constituent elements in the form of pixels with a limited size and has a defined value for each pixel. Digital image representation is 2D matrix with the elements represented by pixel values at each location.[43] To measure the distance between two defined pixels, the Euclidian distance formula can be used as shown in Eq. (1)

Table 3 Hyperparameter testing.

Optimizer	Batch size	Learning rate base	Cosine decay step	Exp decay step	Warmup learning rate	Warmup step	Total loss	Precision IoU = 0.50:0.95	Recall AR@100
SSD MobileNet V2 FPNLite 320 × 320 configuration									
Momentum	12	0.040	100,000	—	0.013	5000	0.074	0.821	0.858
Momentum	**12**	**0.080**	**100,000**	**—**	**0.027**	**5000**	**0.047**	**0.832**	**0.866**
Momentum	12	0.040	100,000	—	0.040	0	0.071	0.823	0.862
Momentum	12	0.040	100,000	—	0.027	5000	0.068	0.814	0.850
RMS_Prop	12	0.004	—	5000	—	—	1.959	0.000	0.036
RMS_Prop	12	0.040	—	500	—	—	0.879	0.389	0.577
SSD ResNet50 V1 FPN 640 × 640 (RetinaNet50) configuration									
Momentum	12	0.040	100,000	—	0.013	5000	0.046	0.821	0.854
Momentum	12	0.080	100,000	—	0.027	5000	0.043	0.793	0.828
Momentum	12	0.040	100,000	—	0.040	0	0.050	0.780	0.813
Momentum	**12**	**0.040**	**100,000**	**—**	**0.027**	**5000**	**0.039**	**0.807**	**0.835**
RMS_Prop	12	0.004	—	5000	—	—	0.578	0.569	0.662
RMS_Prop	12	0.040	—	500	—	—	0.981	0.258	0.547
SSD ResNet101 V1 FPN 640 × 640 (RetinaNet101) configuration									
Momentum	8	0.040	100,000	—	0.013	5000	0.062	0.801	0.835
Momentum	8	0.080	100,000	—	0.027	5000	0.066	0.803	0.831
Momentum	**8**	**0.040**	**100,000**	**—**	**0.040**	**0**	**0.061**	**0.794**	**0.829**
Momentum	8	0.040	100,000	—	0.027	5000	0.053	0.786	0.826
RMS_Prop	8	0.004	—	5000	—	—	0.608	0.599	0.678
RMS_Prop	8	0.040	—	500	—	—	0.726	0.579	0.700

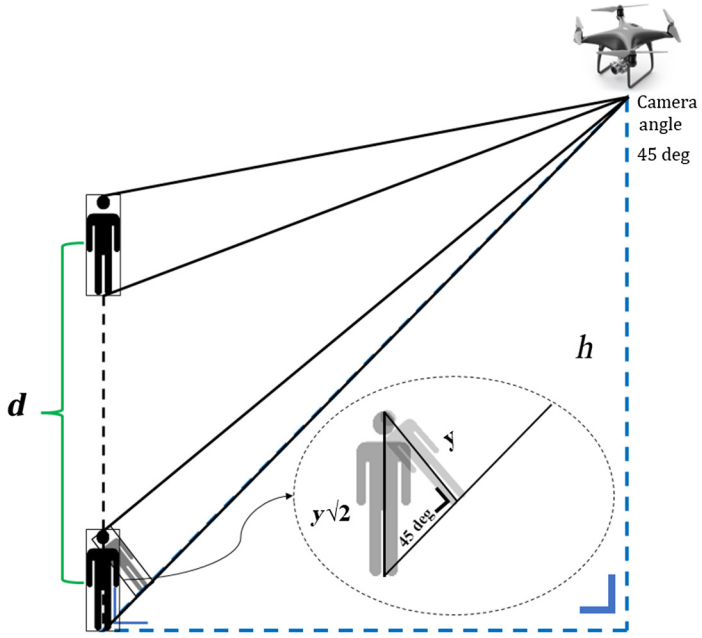


Fig. 7 Illustration of human height captured as a reference for estimating social distance.

$$D_E((i,j),(h,k)) = \sqrt{(i-h)^2 + (j-k)^2}. \tag{1}$$

where (i, j) are the pixels at the starting point and (h, k) are the pixels at the target point.

To estimate the social distance between two or more people, the human height will be used as a basis for measurement. Distance measurement using the human height reference value should be considered further from the point of view of a particular UAV camera. Social distance reference according to human height caught by UAV camera can be seen in Fig. 7.

Using the ratio of height y to get d, the value of d will be generated as shown in Eq. (2)

$$d = \frac{\Delta\text{pixel}_d}{\Delta\text{pixel}_y} \times y, \tag{2}$$

$$y = \frac{y\sqrt{2}}{\sqrt{2}} = \frac{1.6\text{ m}}{\sqrt{2}} \approx 1.13\text{ m}, \tag{3}$$

where d is social distance estimation (meters), Δpixel_d is the number of pixels between two humans that are d apart (pixels), Δpixel_y is the number of pixels that represent the human height from the UAV camera (pixels), and y = human height projection (meters). The height of humans captured by the UAV camera is set at 1.6 m, representing the average height of adults in Indonesia.[44,45] However, this value can be adjusted to accommodate different standard heights across the world. Thus, the human height captured by the camera will be approximated as high as 1.13 m to the projection of the camera angle on the UAV by 45 deg [as shown in Eq. (3)]. The value of y will be substituted into Eq. (2) to estimate the social distance.

3.7 Calibration of Human Coordinate Components Parallel to UAV Camera Viewpoint

The vertical angle of view of the camera determines the human height reference due to the projection on the object that will appear on the detection screen. The flying height of the UAV will be used as a basis in determining the projection to determine the distance of humans who are close to each other from the parallel side of the UAV camera's point of view. Therefore, it is necessary to calibrate the component of human coordinates that are parallel to the point of view of the UAV camera and then substitutes into the Euclidian distance formula in Eq. (1). Illustration of adjusting the projection of human object coordinates parallel to the UAV camera's point of view can be seen in Fig. 8.

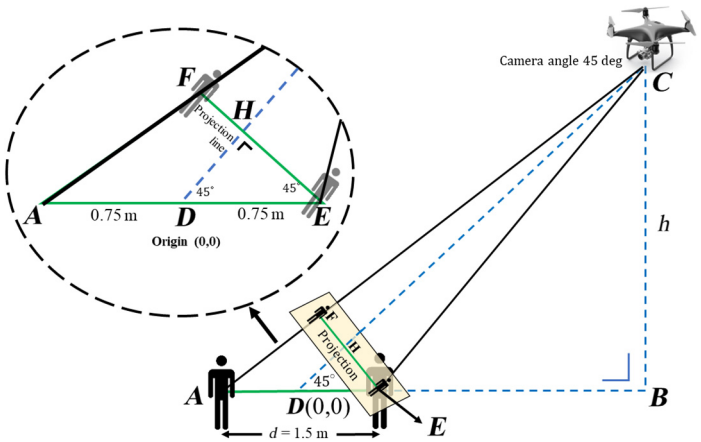


Fig. 8 Illustration of adjusting human object coordinate projection with UAV camera parallel view angle.

The calculation to get the human coordinates projection that is parallel to the camera's point of view is in the letter FE to the AE line. The AE line is 1.5 m long (social distancing requirement) so the AD Line and DE Line have a length of 0.75 m to the center of the camera axis. Since the camera angle has a 45 deg angle to the UAV's maneuverability, the HDE angle also has a 45 deg angle. Since the projection is always perpendicular to the plane on which it is projected, the angles at point H are all 90 deg. Next, the Euclidean equation will be used to determine the social distance for the angle of view that is parallel to the UAV camera while flying, as shown in Eq. (5)

$$d_{\overline{\mathrm{FE}}} = \sqrt{\left(\frac{9}{4(8h+3)} - \frac{3}{4}\right)^2 + \left(\frac{6h}{8h+3}\right)^2}, \tag{4}$$

where $d_{\overline{\mathrm{FE}}}$ is the distance projection estimation (meters), and h is UAV altitude.

It is necessary to calibrate and adjust the value of the y-axis to the flying height of the UAV (h variable) in the coordinates of the estimated social distance. Therefore, the calculation of the new Euclidian formula previously seen in Eq. (1) can be seen in Eq. (6)

$$d_h = \sqrt{(x_1 - x_2)^2 + \left(\frac{\sqrt{\left(\frac{9}{4(8h+3)} - \frac{3}{4}\right)^2 + \left(\frac{6h}{8h+3}\right)^2}}{\frac{3}{2}}(y_1 - y_2)\right)^2}, \tag{5}$$

where d_h is social distance estimation from UAV altitude of h (pixels), x_1, x_2 are the pixel of x in starting point and target point (pixels), and y_1, y_2 are the pixel of y in starting point and target point (pixels).

Euclidian distance in Eq. (6) depends on the value of h variable, so the calibration of the y variable will be different for any given height. The UAV flight test is conducted at a constant altitude so that changes in the value of h will not occur during the program compilation process. To get the original social distance in meters, substitute again in Eq. (2) as $\Delta\mathrm{pixel}_d$.

3.8 Experimental Setup

The object detection test will be conducted with the test dataset, which has a different image set from the train and development dataset. Then, runtime detection test will be conducted on a single image for different resolutions. One of the best models will be selected which will then be implemented in the program to display the results of human object detection with the approximate distance between the human objects. The pilot will turn on the UAV along with the remote control which is the main component in the crowd detection system. The mobile phone will be connected to the remote control as a tool in handling video matters. The pilot will maneuver the UAV and search for crowd points via video transmitted from the UAV camera via the Wi-Fi direct system. The input video is not processed directly on the UAV but is done on a local computer in real time. This processing method is to overcome the UAV specifications that are not able to

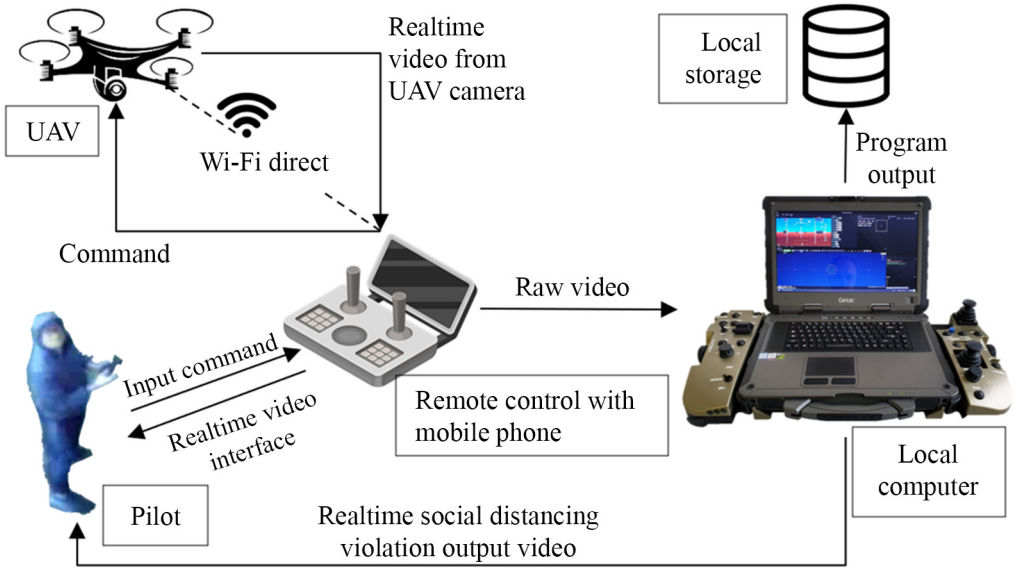


Fig. 9 Architecture diagram of crowd detection system testing.

embed a companion computer (microprocessor). Furthermore, evaluating the entire human detection system can be seen in Fig. 9.

The UAV must be in a radius of <50 m from the remote-control location to avoid loss of contact. After the UAV has successfully flown at a certain altitude, the mobile phone will send raw video to a local computer that contains a program to calculate the number of social distancing violations. The result of the violation will be displayed on the pilot and first stored on local storage. The testing process will keep looping until the UAV has landed perfectly and program execution has been stopped. To minimize detection delays, a detection will be performed for each specific frame interval. For every N frame, one detection will be performed, including the measurement of the distance between humans and other operations. This interval setting is also useful for getting objects with other positions that may be caught on camera, so time does not run out just to detect at the same position. In addition, the memory used to store the image of the violator will also be less, so the memory can be used for other purposes.

3.9 Hardware and Resources

The UAV, which is used for crowd video recording, is the TXD 8S(L) Drone Wi-Fi HD Camera. For training phase, Google Colab provides a single 12 GB NVIDIA Tesla K80 GPU that can be used up to 6 h continuously. The program development and testing phase are using a computer with Intel® Core™ i7-9750H CPU @ 2.60 GHz, with 8.192 MB RAM.

4 Results and Discussion

4.1 Crowd Detection Model Test Results

There are three test metrics conducted at the final evaluation stage of the model, depicted by a bar graph as can be seen in Fig. 10.

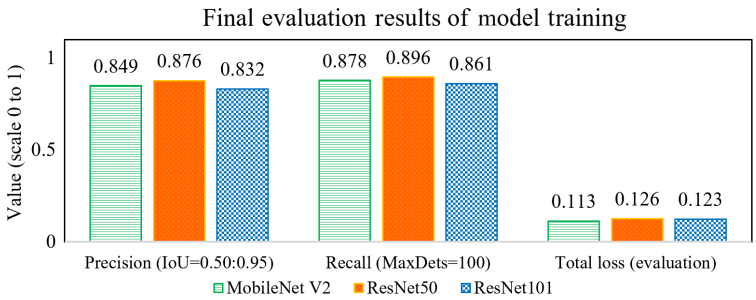


Fig. 10 Graph of final evaluation results of model training.

The intersection over union (IoU) metric represents how big the wedge area is between the detection box on ground truth and the detection box formed from the prediction data. The equation of the IoU metric can be seen in Eq. (6)

$$\mathrm{IoU}(p,a) = \frac{(\mathrm{Box}_T \cap \mathrm{Box}_P)}{(\mathrm{Box}_T \cup \mathrm{Box}_P)}, \tag{6}$$

where $\mathrm{IoU}(p,a)$ is introduction over union [0.1], Box_T = pixel of x in starting point and target point (pixels). Meanwhile, Box_P is the pixel of y in starting point and target point (pixels).

All human crowd detection models yield more than 80% precision and recall values. In addition, all human detection models also have a low loss value, which is below the 0.2 scale. Precision metric is used to determine the ratio between the correct prediction data detection boxes compared to the overall detection results as shown in Eq. (7). In addition, the recall metric is used to determine the ratio between the correct prediction data detection boxes compared to the overall label data that should be formed (ground truth) as shown in Eq. (8)

$$\mathrm{Precision} = \frac{\mathrm{True\ positive}}{\mathrm{True\ positive} + \mathrm{False\ positive}}, \tag{7}$$

$$\mathrm{Recall} = \frac{\mathrm{True\ positive}}{\mathrm{True\ positive} + \mathrm{False\ negative}}. \tag{8}$$

True positive is positive class detection results being in accordance with the basic truth. False positive is a detection result being a positive class, but the basic truth should be a negative class. False negative is detection result being a negative class, but the ground truth should be a positive class.

Precision value exceeding 80% indicates that the number of true positives is 4 times greater than the number of false positives [Eq. (7)]. That is, the number of images detected as "human" corresponds to the ground truth of the category, which is also "human." Only <20% of "human" images were incorrectly detected as "non_human." Furthermore, if the recall value exceeds 80%, the number of true positives is 4 times greater than the number of false negatives [Eq. (8)]. This indicates that the number of images detected are of the "human" class and identical with the basic truth, which is also "human" category. Less than one fifth of the image is detected as "non_human" even though it has the basic truth of "human." In determining true positive and false negative, IoU will be involved for certain value limits. If the IoU between the predictive data detection box and the ground truth is higher than 0.5, the area can be defined as TP. Otherwise, it will be FP.[46]

Regarding the loss value, which has a scale of <0.2, there are 2 variables that have an impact on the total loss calculation. This loss function is impacted by the classification loss value and the localization loss value

$$L(x,c,l,g) = \frac{1}{N}(L_{\mathrm{conf}}(x,c) + \alpha L_{\mathrm{loc}}(x,l,g)), \tag{9}$$

where N is the number of detection boxes in a detection, x is image to be detected, c is the predictive value of the predictive confidence, l is the detection box formed from the results of object detection, and g is label data detection box for ground truth.

As seen in Eq. (9), the error caused in the category classification phase is ridiculously small. The low value of classification loss can also be caused by a high detection confidence level. On the other hand, the low value of localization loss is supported by the appearance of a detection box that matches the ground truth coordinates that have been defined in the image dataset in XML format. The best precision and sensitivity values were obtained by the ResNet50 model, with a precision value reaching 87.6% and a sensitivity (recall) value reaching 89.6%. However, the lowest loss value among all crowd detection models is obtained from the MobileNet V2 metric model, which is 0.113. However, MobileNet is a crowd detection model with the smallest size compared to the other 2 models, only having a size of 27 MB (Fig. 11).

Furthermore, the testing of the human crowd detection model was conducted by entering two images, namely the first image containing humans and the second image containing motorcycle riders. The results of the runtime testing of each model can be seen in Table 4.

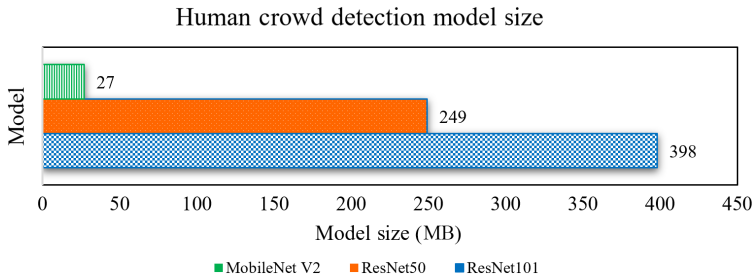


Fig. 11 Human crowd detection model size chart.

Table 4 Human detection runtime test results.

Model	Resolution	Load time (s)	Image #1 runtime (s)	Image #2 runtime (s)	Detection runtime average (s)
MobileNet V2	480 × 360	7.387	1.044	0.117	0.581
	640 × 480	6.863	1.099	0.119	0.609
	960 × 720	7.471	1.101	0.129	0.615
	1440 × 1080	7.286	1.102	0.141	0.621
	Average	7.252			0.606
ResNet50	480 × 360	7.201	1.478	0.389	0.934
	640 × 480	7.515	1.501	0.392	0.947
	960 × 720	9.123	1.685	0.430	1.058
	1440 × 1080	9.438	1.701	0.444	1.073
	Average	8.319			1.003
ResNet101	480 × 360	12.472	1.936	0.541	1.239
	640 × 480	13.685	1.972	0.541	1.256
	960 × 720	13.748	2.192	0.615	1.403
	1440 × 1080	15.022	2.207	0.661	1.434
	Average	13.732			1.333

Model setup time is the time measured during the model initiation phase until the model is ready for use. Detection time is the time measured when the human crowd detection module is run until the output is the coordinates of the detection box and the number of objects detected (Fig. 12). The duration of image processing is very dependent on the size of the image resolution received by the human crowd detection model. The larger the resolution size of the human crowd image, the greater the human detection time. For this reason, it is necessary to pay attention to the size of the input video resolution when using the human crowd detection model if it has a time constraint.

(a) (b)

Fig. 12 Example of (a) "human" detection and (b) "non_human" detection result.

Considering the results of precision values, sensitivity, total loss, and model size, MobileNet V2 was chosen as the best human crowd detection model. Furthermore, the testing of the social distance estimation module and the program for calculating the number of social distancing violations will use the MobileNet V2 model.

4.2 Social Distance Estimation and Calibration Test Results

Furthermore, the test is conducted at different altitude. The flying altitude of the UAV is used as one of the factors to determine social distance calibration and has an indirect impact on the sensitivity of the human detection model. Social distance estimation is conducted for every two people detected in proximity. An example of an image from the estimation of social distance at a height of ±5 m can be seen in Fig. 13. The lines of social distancing violations are interconnected with each other. For this reason, it is necessary to calculate the number of violators and the number of social distancing violations to find out the number of people that participate in the crowd. Calibration will make a correction to the difference in the angle of the UAV camera while flying. By calibrating, the excess distance caused by the illusion of a viewing angle can be overcome as shown in Eq. (6).

4.3 Evaluating the Program for Calculating the Number of Social Distance Violations

In general, there are several mandatory conditions that need to be met to fulfill the requirements of detecting human social distancing violations, as shown in Fig. 14. First, the estimated social

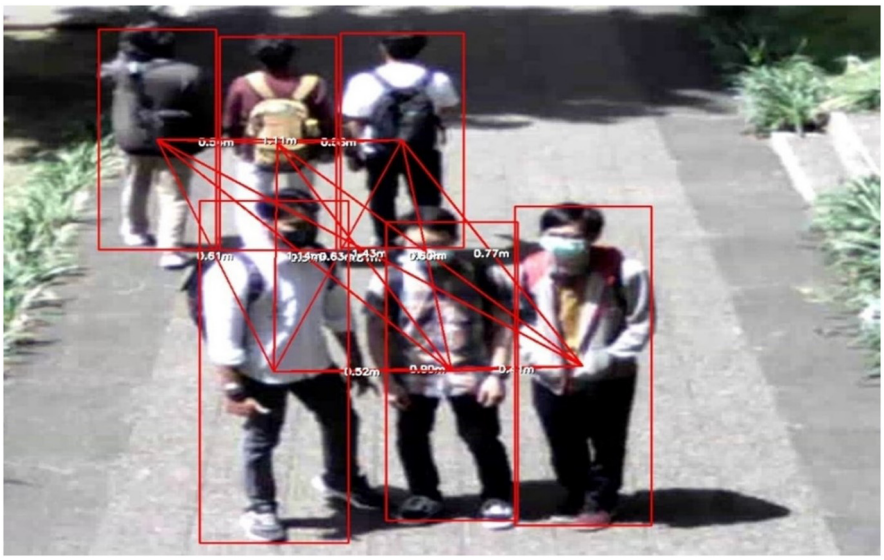


Fig. 13 Example of the estimation of social distance at a height of ±5 m.

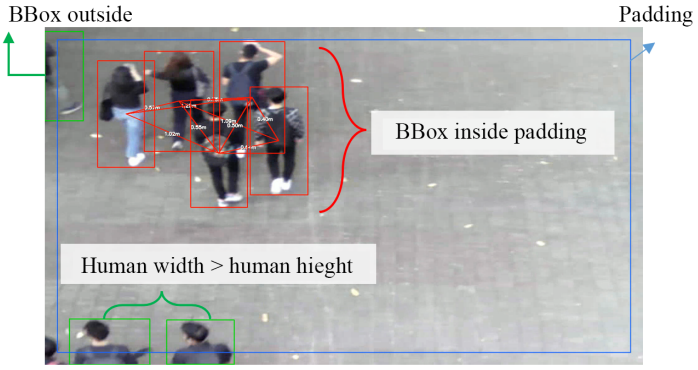


Fig. 14 Mandatory requirements conditions for detecting social distance violations.

distance must be <1.5 m, which is measured based on the original distance obtained from the calibration results using the UAV camera's viewpoint as described in Eq. (6). Second, the detection box should fall within the defined padding limits, determined by the offset numbers of the x and y components. This requirement ensures that the standard height of humans is not compromised by objects that may intersect with the image frame. The width of the detection box should be smaller than its height to minimize errors when measuring social distance estimates in humans.

With the formation of a crowd of people in one place, the potential for social distancing violations is extremely high. The program will be evaluated to ensure that the integration between the human detection module, the social distance estimation and calibration module, and the social distance enumeration module is not a problem. Human crowd video in MP4 format will be input to the program and produce video output in MP4 format, as seen in the architecture of the crowd detection system testing system (Fig. 9). An example of the calculation results of violators and social distancing violations can be seen in Fig. 15.

The program succeeded in calculating the number of violators and social distancing violations. As seen in Fig. 15, five humans were detected as social distancing violators. In addition, there are also five social distancing violations for every two humans who are close together. Because all human objects are within the padding limit, the total human height has been calculated and can be used as a reference to calculate social distance.

The first baseline paper test will use the same image dataset, namely the Oxford Town Center, whereas the second test will use the human dataset taken from ITB area. Using the Oxford Town Center dataset,[47] this paper has advantages compared to previous studies (see Table 5).

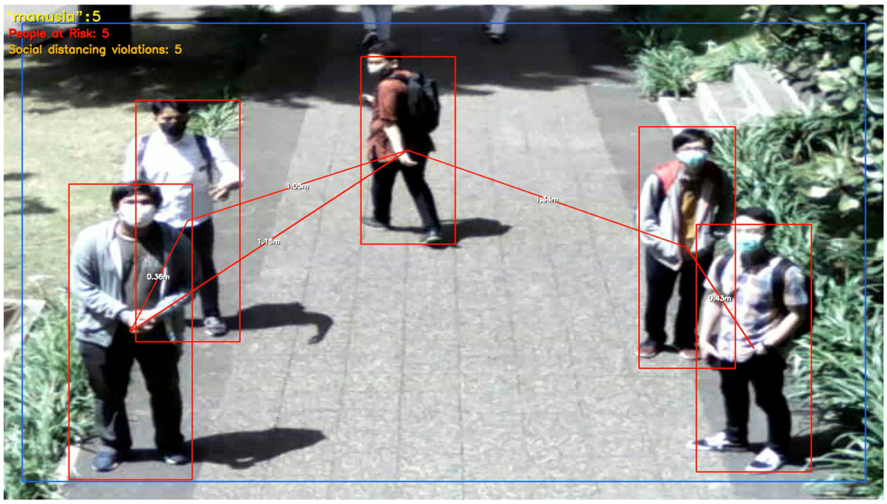


Fig. 15 Example of violation calculation results (red text) and social distance violation (orange text).

Table 5 Comparison between this paper and other previous studies on Oxford Town Center dataset.

Research	Backbone	Precision (%)	Measured social distance	Counting social distancing violators
Rezaee et al.[15]	ShuffleNet	88.45	—	—
Elbishlawi et al.[48]	DETR + ResNet50	43.4	—	—
Özbek et al.[49]	Darknet-53 + YOLOv3	55.3	✓	—
Madane and Chitre[50]	ResNet50	94.23	—	—
Ahmad et al.[51]	YOLOv3	97	✓	—
Wastupranata and Munir **(Proposed)**	MobileNet V2	82	✓	✓

Table 6 Comparison with other research using human datasets in ITB area.

Research	Model	Precision (%)	Size (MB)
Howard et al.[52]	MobileNet V2	40.88	76.3
He et al.[53]	ResNet50	71.70	261
He et al.[53]	ResNet101	74.21	411
He et al.[53]	ResNet152	74.21	537
Tan et al.[54]	EfficientDet	93.08	63.7
Zhou et al.[55]	CenterNet	91.82	1480
Wastupranata and Munir (Proposed)	MobileNet V2	96.86	26.6

The model precision values in this paper are higher than the detection transformer (DETR) + ResNet50 and Darknet-53 + YOLOv3 backbones, but slightly lower than ShuffleNet, ResNet50, and YOLOv3. The purpose of this paper is to find a model with a load that can be handled by a UAV companion computer. It can also be seen that social distance measurements were also carried out in the study of Özbek et al.[49] and Ahmad et al.[51] This paper has integrated the measurement of social distance estimation with the enumeration of social distance violators, which has not been implemented by the other five studies.

The results of a comparison of crowd detection systems using human image dataset at the ITB can be seen in Table 6. The model precision value in this final project is the highest compared to other pre-trained models. The next good precision value is found in the EfficientDet model, reaching 93.08%. The interesting thing is that MobileNet V2 before training has a bad precision value, which is only 40.88%.

Comparison of human detection models will be seen from the size of each. The model in this final project has the lowest model size, around 26.6 MB, lower than the MobileNet V2 model, which has not been trained (76.3 MB). This is because the pre-training model must store 90 different class categories. CenterNet has the largest size, which is 1.48 GB. The model in this paper is suitable for detecting crowds of people using the image dataset at the ITB with a small model size and very high precision values.

5 Conclusions

Three human detection models were successfully created using the MobileNet, ResNet50, and ResNet101 pre-trained models. All models can detect humans, cyclists, and motorcyclists with precision and sensitivity values above 80%. All trained models also did not experience overfitting during training, as evidenced by the loss function value below the 0.2 scale. MobileNet V2 was chosen as the detection model for further implementation in the social distance calculation program. This is because the MobileNet V2 model has a file size only 19 MB, so the detection process can be conducted smoothly according to the computational load that can be managed by the UAV companion computer. The precision value of MobileNet V2 reaches 84.9% (IoU = 0.50:0.95), with a sensitivity value (recall) reaching 87.8% (MaxDets = 100).

The estimation of social distance was successfully conducted using the average human height in Indonesia as a reference, which is 1.6 m. The social distance calibration formula for the social distance component that is parallel to the UAV camera's point of view has been successfully implemented in the program so the estimated social distance is close to the original distance. However, the flying height of the UAV must be determined in advance so the estimated social distance can be properly calibrated. The social distancing violation calculation program has been successfully integrated with the crowd detection model and the social distance estimation and calibration calculation module.

6 Future Works

In the future, an improved architectural model will be conducted that can detect crowds of people more quickly. In addition, hyperparameter tuning can be done with other variables to increase the accuracy and sensitivity of the resulting model. Furthermore, the number of images for model training should be increased, so the UAV can detect human crowds in a more heterogeneous environment. Thus, the program can be further developed to conduct human tracking so the movements of social distance violators can be further traced. The measurement of social distance will be developed using proximity sensors that are integrated with the UAV companion computer. The UAV's flight altitude measured from the ground can also be determined using the proximity sensor. It is also possible to develop a crowd detection model in darker places, using a thermal sensor. The detection of social distancing violators can also be conducted on humans with temperatures higher than the normal reference so that it can be seen whether the human being is suspected of being a COVID-19 suspect.

References

1. M. Qian and J. Jiang, "COVID-19 and social distancing," *Can. J. Addict.* **11**, 4–6 (2020).
2. B. J. Cowling et al., "Impact assessment of non-pharmaceutical interventions against coronavirus disease 2019 and influenza in Hong Kong: an observational study," *Lancet Public Health* **5**, e279–e288 (2020).
3. M. Greenstone and V. Nigam, "Does social distancing matter?" SSRN Electron. J. (2020).
4. IHME, "Forecasting COVID-19 impact on hospital bed-days, ICU-days, ventilator-days and deaths by US state in the next 4 months," MedRxiv (2020).
5. J. K. Lee and Y. J. Choe, "The impact of social distancing on the transmission of influenza virus, South Korea, 2020," *Osong Public Health Res. Perspect.* **12**, 91–92 (2021).
6. R. I. Kemenkes, *Protokol Pemicuan dan Verifikasi 5 Pilar STBM* (2020).
7. J. Xu et al., "DGG: a novel framework for crowd gathering detection," *Electronics* **11**, 31 (2022).
8. A. H. Ozcan, C. Unsalan, and P. Reinartz, "Sparse people group and crowd detection using spatial point statistics in airborne images," in *RAST 2015 - Proc. 7th Int. Conf. Recent Adv. Sp. Technol.*, pp. 307–310 (2015).
9. M. Al-Sa'd et al., "A social distance estimation and crowd monitoring system for surveillance cameras," *Sensors* **22**, 1–21 (2022).
10. R. Krishna, *Computer Vision: Foundations and Applications*, Stanford University (2017).
11. R. Venkatesan and B. Li, *Convolutional Neural Networks in Visual Computing: A Concise Guide*, CRC Press (2017).
12. R. Eshel and Y. Moses, "Homography based multiple camera detection and tracking of people in a dense crowd," in *26th IEEE Conf. Comput. Vis. Pattern Recognit., CVPR* (2008).
13. D. K. Singh et al., "Human crowd detection for city wide surveillance," *Proc. Comput. Sci.* **171**, 350–359 (2020).
14. C. Papaioannidis, I. Mademlis, and I. Pitas, "Autonomous UAV safety by visual human crowd detection using multi-task deep neural networks," in *IEEE Int. Conf. Rob. Autom. (ICRA 2021)*, pp. 11074–11080 (2021).
15. K. Rezaee et al., "An autonomous UAV-assisted distance-aware crowd sensing platform using deep shufflenet transfer learning," *IEEE Trans. Intell. Transp. Syst.* **23**(7), 9404–9413 (2021).
16. Z. Shao et al., "Real-time and accurate UAV pedestrian detection for social distancing monitoring in COVID-19 pandemic," *IEEE Trans. Multimedia* **24**, 2069–2083 (2021).
17. M. Butenuth et al., "Integrating pedestrian simulation, tracking and event detection for crowd analysis," in *Proc. IEEE Int. Conf. Comput. Vis.*, pp. 150–157 (2011).
18. Y. Lecun, Y. Bengio, and G. Hinton, "Deep learning," *Nature* **521**, 436–444 (2015).
19. N. Dalal and B. Triggs, "Histograms of oriented gradients for human detection," in *Proc. - IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit., CVPR 2005*, Vol. I, pp. 886–893 (2005).
20. Y. Ma et al., "Pedestrian detection and tracking from low-resolution unmanned aerial vehicle thermal imagery," *Sensors* **16**, 446 (2016).
21. S. Roth, "People-tracking-by-detection and people-detection-by-tracking," in *IEEE Conf. Comput. Vis. and Pattern Recognit.*, Anchorage, Alaska, United States, pp. 1–8 (2008).
22. J. Choi, B.-J. Lee, and B.-T. Zhang, "Human body orientation estimation using convolutional neural network," https://doi.org/10.48550/arXiv.1609.01984 (2016).
23. M. A. Ansari and D. K. Singh, "Human detection techniques for real time surveillance: a comprehensive survey," *Multimedia Tools Appl.* **80**, 8759–8808 (2021).

24. A. Lohia et al., "Bibliometric analysis of one-stage and two-stage object detection," *Libr. Philos. Pract.* **4910**, 34 (2021).
25. W. Liu et al., "SSD: single shot multibox detector," *Lect. Notes Comput. Sci.* **9905**, 21–37 (2016).
26. Y. S. Tsai, A. V. Modales, and H. T. Lin, "A convolutional neural-network-based training model to estimate actual distance of persons in continuous images," *Sensors* **22**, 5743 (2022).
27. M. A. Rahman and M. Hamada, "PCBMS: a model to select an optimal lossless image compression technique," *IEEE Access* **9**, 167426–167433 (2021).
28. Tzutalin, "LabelImg," Git code. https://github.com/heartexlabs/labelImg (2015).
29. M. Everingham et al., "The PASCAL visual object classes (VOC) challenge," *Int. J. Comput. Vis.* **88**, 303–338 (2010).
30. TensorFlow, "TensorFlow 2 detection model zoo," GitHub. https://github.com/tensorflow/models/blob/master/research/object_detection/g3doc/tf2_detection_zoo.md (2022).
31. X. Li et al., "Feature pyramid networks for object detection," in *Proc. - 2019 IEEE Int. Conf. Parallel Distrib. Process. with Appl. Big Data Cloud Comput. Sustain. Comput. Commun. Soc. Comput. Netw., ISPA/BDCloud/SustainCom/SocialCom 2019*, pp. 1500–1504 (2019).
32. M. K. Makirin, L. M. Wastupranata, and A. Daffa, "Onboard visual drone detection for drone chasing and collision avoidance," *AIP Conf. Proc.* **2366**, 060013 (2021).
33. L. M. Wastupranata and R. Munir, "UAV detection using web application approach based on SSD pre-trained model," in *Proc. IEEE Int. Conf. Aerosp. Electron. Remote Sens. Technol. ICARES* (2021).
34. J. Huang et al., "Speed/accuracy trade-offs for modern convolutional object detectors," in *Proc. - 30th IEEE Conf. Comput. Vis. Pattern Recognit., CVPR*, January, pp. 3296–3305 (2017).
35. L. N. Smith, "A disciplined approach to neural network hyper-parameters: Part 1—learning rate, batch size, momentum, and weight decay," https://doi.org/10.48550/arXiv.1803.09820 (2018).
36. I. Loshchilov and F. Hutter, "SGDR: stochastic gradient descent with warm restarts," in *5th Int. Conf. Learn. Represent., ICLR 2017 – Conf. Track Proc.*, pp. 1–16 (2017).
37. J. G. López, "Geometric computer vision meets deep learning for autonomous driving applications," https://dialnet.unirioja.es/servlet/tesis?codigo=300481 (2020).
38. A. Gotmare et al., "A closer look at deep learning heuristics: learning rate restarts, warmup and distillation," in *7th Int. Conf. Learn. Represent. ICLR* (2019).
39. P. Goyal et al., "Accurate, large minibatch SGD: training ImageNet in 1 hour," https://doi.org/10.48550/arXiv.1706.02677 (2017).
40. N. H. Phong, A. Santos, and B. Ribeiro, "PSO-convolutional neural networks with heterogeneous learning rate," *IEEE Access* **10**, 89970–89988 (2022).
41. S. Ruder, "An overview of gradient descent optimization algorithms," https://doi.org/10.48550/arXiv.1609.04747 (2016).
42. R. F. Lyon, "Neural networks for machine learning," in *Human and Machine Hearing*, pp. 419–440, Cambridge University Press, Cambridge (2017).
43. M. Sonka, V. Hlavac, and R. Boyle, *Image Processing, Analysis, and Machine Vision*, Cengage Learning (2014).
44. A. B. Pulungan et al., "Indonesian national synthetic growth charts," *Acta Sci. Paediatr.* **1**, 1–15 (2018).
45. H. A. Puswadi and S. Sunyoto, "Rancang Bangun Alat Pengering Bahan Makanan Berbasis Wings Drying System Dengan Dua Sumber Panas," *J. Ilm. Teknosains* **7**, 36–43 (2021).
46. H. Shen et al., "Priority branches for ship detection in optical remote sensing images," *Remote Sens.* **12**(7), 1206 (2020).
47. B. Benfold and I. Reid, "Stable multi-target tracking in real-time surveillance video," in *Proc. IEEE Conf. Comput. Vis. and Pattern Recognit.*, IEEE, pp. 3457–3464 (2011).
48. S. Elbishlawi, M. H. Abdelpakey, and M. S. Shehata, "SocialNet: detecting social distancing violations in crowd scene on IoT devices," in *7th IEEE World Forum Internet Things, WF-IoT 2021*, pp. 801–806 (2021).
49. M. M. Özbek, M. Syed, and I. Öksüz, "Subjective analysis of social distance monitoring using YOLO v3 architecture and crowd tracking system," *Turk. J. Electr. Eng. Comput. Sci.* **29**, 1157–1170 (2021).
50. S. Madane and D. Chitre, "Social distancing detection and analysis through computer vision," in *6th Int. Conf. Converg. Technol. I2CT 2021*, pp. 1–10 (2021).
51. I. Ahmad et al., "Analytical study of deep learning-based preventive measures of COVID-19 for decision making and aggregation via the RISTECB model," *Sci. Program.* **2022**, 6142981 (2022).
52. A. G. Howard et al., "MobileNets: efficient convolutional neural networks for mobile vision applications," https://doi.org/10.48550/arXiv.1704.04861 (2017).
53. K. He et al., "Deep residual learning for image recognition," in *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit.*, pp. 770–778 (2016).
54. M. Tan, R. Pang, and Q. V. Le, "EfficientDet: scalable and efficient object detection," in *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit.*, pp. 10778–10787 (2020).
55. X. Zhou, D. Wang, and P. Krähenbühl, "Objects as points," https://doi.org/10.48550/arXiv.1904.07850 (2019).

Leonard Matheus Wastupranata received his bachelor's degree in informatics engineering from the Institut Teknologi Bandung, Indonesia, in 2023. His research interests include object detection, deep learning, computer vision, image processing, and autonomous aerial vehicles.

Rinaldi Munir received his bachelor's degree in informatics engineering and his MSc degree from Institut Teknologi Bandung (ITB), Indonesia. He earned his PhD in 2010 from the School of Electrical Engineering and Informatics, ITB. He has been a lecturer in ITB's Informatics Department since 1993 and is currently an associate professor in the School of Electrical Engineering and Informatics. His research interests include cryptography, steganography, digital image processing, fuzzy logic, and numerical computation.